Cahier des charges - Projet Info-SPE 2006-07

mardi 23 janvier 2007

Cadre

Le projet est à réaliser en groupe de quatre personnes (et seulement quatre). Sa durée est d'environ quatre mois (de Février à Juin). Ce cahier des charges présente la nature du projet, les différentes parties qui le composent et les délais de réalisation (planning).

Le sujet

Le projet que vous allez devoir réaliser, est un logiciel de reconnaissance de caractères : un O.C.R. pour Optical Character Recognition. Il va permettre de récupérer dans un traitement de texte des documents imprimés (courriers, pages de livre, articles de presse). Après numérisation, le logiciel attribue à chaque symbole du texte le caractère correspondant à partir d'un profil entraîné avec des exemples. Mais la transformation en texte brut d'une page numérisée n'est pas la seule fonctionnalité du logiciel. Il analyse également le mot reformé avec un dictionnaire pour diminuer le nombre d'erreurs.

Restrictions

Ce projet sera développé sous NetBsd et sera codé en C et Objective Caml. Ce projet, en plus d'être techniquement intéressant, permettra, comme le sujet est unique pour tous les groupes, de se positionner par rapport aux autres et d'établir un vrai challenge quant à sa réalisation et ses performances.

Protocole

Vous devez choisir un groupe de quatre personnes, donner un nom à ce groupe, choisir un sujet de projet et remettre le nom du groupe, le nom des membres, le login de chaque membre, le nom du projet ainsi que le nom de celui que vous aurez désigné comme chef de projet (cf. date limite de choix de groupe). Un chef de projet est nécessaire pour servir, entre autres, d'interlocuteur aux jurys des soutenances.

Les soutenances

Les soutenances durent, à l'exception de la dernière, 15 minutes. Elles sont au nombre de trois. Elles se suivent et donc tout retard de votre part entraîne une diminution de votre temps de parole! Il faut donc faire une présentation claire, nette et précise de l'état d'avancement de votre projet et répondre de la manière la plus claire et la plus concise possible aux questions du jury. Les deux premières soutenances auront lieu en soirée (à partir de 17h30).

Dates des soutenances

Première : du 5 au 9 Mars 2007

Deuxième : du 2 au 6 Avril 2007

Finale : du 28 au 31 Mai 2007

Le jury

Les soutenances seront assurés pour les deux premières par les Yakas avec éventuellement un enseignant. La dernière (soutenance finale) sera effectuée par un enseignant assisté des Yakas vous ayant suivis lors des précédentes soutenances.

Bonus

Bien évidemment, des ajouts et/ou modifications pourront être effectués en cours d'année à ce cahier des charges. Détection d'images, de photos, de nouveaux formats de typographie, etc. La détection de police manuscrite particulièrement complexe. L'imposition d'utilisation de certaines bibliothèque de programme, etc.

1 Nature et but de ce projet

Qu'est-ce que ?
C'est un logiciel de reconnaissance de caractères manuscrits et imprimés à apprentissage automatique. Si vous possédez un scanner, il vous suffit de scanner l'image et le logiciel le transcrira pour vous en texte mis en forme.

Principe :
Le principe est simple : L'utilisateur fournit au logiciel plusieurs documents A4 numérisés contenant la liste des symboles à reconnaître (son écriture), et ce en plusieurs exemplaires, ainsi que la traduction de ces symboles en texte ASCII. Le logiciel utilise cette base d'informations pour l'apprentissage d'un profil utilisateur.

Ensuite, lorsque l'utilisateur veut convertir un document numérisé (sous forme d'image) en texte, il ouvre simplement le document dans le logiciel et lance la procédure de conversion. Le logiciel donne alors l'équivalent du texte mis en forme.

Règles à respecter :
Pour que le développement de ce projet se passe le mieux possible, nous allons respecter les règles de codage suivantes :

- ***Pour les parties en OCaml :***
  - Indentation des fonctions obligatoire.
  - Pas plus de 40 lignes par fonction.
  - Les noms de fonctions, variables, constantes et macros devront être en anglais.
  - Mettre un maximum de commentaires.
  - Établir une hiérarchie correct et cohérente des modules ne contenant pas de cycle.
  - Les modules devront être utilisé avec la notation pointé (`Module.fonction`) ; la directive `open` est prohibée.
  - Pour chaque fichier `.ml`, vous devrez faire un fichier `.mli`.
  - Pour chaque module interne vous devrez fournir sa signature.
  - Pas plus de 3 niveaux d'imbrication de modules.
  - 80 caractères par ligne.
  - Pas d'espace en fin de ligne dans les fichiers `.ml` et `.mli`
  - Aucun warning à la compilation.

- ***Pour les parties en C :***
  - Indentation des fonctions obligatoire.
  - Pas plus de 40 lignes par fonction.
  - Pas de variable globale.
  - Les noms de fonctions, variables, constantes et macros devront être en anglais.
  - Mettre un maximum de commentaires.
  - Faire une hiérarchie cohérente des fichiers `.c`.
  - Pour chaque fichier .c faire un fichier `.h`.
  - 80 caractères par ligne.
  - Pas d'espace en fin de ligne dans les fichiers `.c` et `.h`.
  - Aucun warning à la compilation avec les drapeaux de compilation suivants : `-Wall -W -pedantic`.
- ***Pour le projet :***
  - Votre projet doit disposer d'un `Makefile` compatible avec *GNU Make*.
  - Votre `Makefile` devra fournir une règle `opt` pour construire un binaire natif dont le nom sera suffixé par `.opt` et une règle `byte` pour une construire un binaire en bytecode. Par défaut la règle `all` produit les deux types de binaires.
  - Votre `Makefile` devra contenir une règle `clean` pour effacer tous les produits de la compilation.
  - Un fichier `README` qui explique comment compiler et utiliser votre projet.

2 Découpage du projet

Pour effectuer un partage des tâches, le découpage des tâches sera fait de la façon suivante :

1. Pré-traitement de l'image de départ, numérisée à partir d'un document A4, par l'effacement de bruits afin de produire un document "parfait".
2. Extraction des lignes, des caractères, et détection de la mise en forme du document (nous nous contenterons, dans un premier temps de la typographie des caractères (gras, italique, souligné, etc.).
3. Codage des caractéristiques d'une image d'un caractère, et apprentissage machine des caractéristiques de chaque symbole à détecter.
4. Développement de l'interface visuelle de l'application, interprétation et affichage de la mise en forme du document détecté antérieurement et définition de balises de mise en forme du document texte (le format des balises est le html).

2.1 Pré-traitement de l'image *(en C)*

Concrètement, il s'agira de transformer l'image d'origine en la faisant pivoter dans le cas où l'image n'est pas droite (pour une reconnaissance ultérieure plus facile des caractères) et de faire ressortir le tracé des caractères, c'est-à-dire : Gommer le bruit de fond et amplifier les marques des caractères.

Il faudra donc envisager différentes méthodes de rotation d'images et de détection d'inclinaison du texte. Il faudra aussi trouver une méthode pour discerner le bruit de fond des caractères.

Dans le cas idéal, l'image qui sortira de ce traitement sera un fichier bitmap sur lequel les caractères seront en noir sur fond blanc et disposés horizontalement. Selon l'état des images fournies à traiter, il sera difficile, ou plutôt inutile, de vouloir un fond absolument blanc. Le traitement de l'image dans cette partie étant global, il ne rentre pas dans le détail du document : Certaines tâches ne seront donc supprimées que lors d'une analyse plus pointue de l'image.

Par exemple, une tâche sur la feuille qui masquerait plusieurs caractères ne pourra pas être détectée lors de cette première phase du projet mais uniquement lors de l'extraction des caractères ou de leur interprétation.

2.2 Analyse de l'image et extraction des caractères *(en OCaml)*

Cette partie fonctionne à partir d'un document "parfait" issu de la phase de pré-traitement de l'image. Elle est essentielle puisqu'elle va fournir les caractères découpés au système de reconnaissance. Il faudra donc créer des algorithmes capables de détecter des lignes et des caractères et de stocker les coordonnées en mémoire d'une manière assez fonctionnelle pour qu'elle soit exploitable simplement.

Les algorithmes devront être bien pensés et la structure modulable en cas de modification.

Le principe est simple : Le document va isoler les pixels foncés et clairs pour situer globalement le texte sur l'image et repérer dans un premier temps les lignes. Ensuite il faudra détecter les bornes hautes et basses de chaque ligne. Une fois cette étape réalisée, on fera une détection des caractères en les encadrant par la largeur, et grâce aux bornes de la ligne, on isolera plus précisément les caractères en hauteur... .

Ensuite toutes ces informations stockées seront envoyées au système de reconnaissance des caractères et serviront également à la mise en page du texte. Il faudra aussi tester un maximum d'exemples pour détecter les cas qui ne fonctionneraient pas et les modifier pour obtenir la meilleure détection possible.

2.3 Reconnaissance des caractères *(en OCaml)*

Il a été décidé que la reconnaissance s'effectue par apprentissage automatique. La reconnaissance des caractères utilisera comme technique d'apprentissage les réseaux de neurones. Le travail consistera d'abord à réaliser la partie d'apprentissage de chaque symbole à détecter, et ensuite celle effectuant la

classification des symboles à convertir en texte. Ce travail se fera sur des symboles déjà isolés par la partie *Analyse de l'image et extraction des caractères.*

Mais la réalisation du réseau de neurones ne sera pas la seule tâche à effectuer, il faudra également trouver une bonne représentation des données caractérisant l'image d'un symbole, et standardiser ces données d'entrée pour le réseau de neurones.

Il faudra également gérer le cas des caractères trop ambigus ou trop difficiles à reconnaître. Le "classifieur" devra les rejeter pour pouvoir ensuite les traiter au niveau de l'interface.

2.4 Interface graphique *(en C ou OCaml)*

Une interface graphique, quoique non nécessaire, reste néanmoins une façon intuitive et rapide d'utiliser un logiciel.

Cette interface sera relativement simple : Une zone de texte afin de permettre à l'utilisateur de modifier le texte du document numérisé prendra la moitié de la fenêtre. L'autre moitié étant consacrée à la visualisation du document original afin de pouvoir vérifier la similarité des textes.

3 Interface *(en C ou OCaml)*

Ce projet est à réaliser en C et en OCaml. Le découpage entre les deux langages doit suivre les recommandations suivantes :

- Le chargement, la sauvegarde et les manipulations des fichiers sont à réaliser en C.
- Les pré-traitements (et toutes les manipulations d'images) seront à réaliser en C.
- L'interface graphique est à réaliser à l'aide de la bibliothèque graphique GTK. L'usage de la version OCaml de la bibliothèque GTK (*lablgtk*) est autorisée.
- Vos principaux algorithmes devront être implantés en OCaml, en particulier vous devrez établir une interface de communication claire entre le chargement et la sauvegarde des données (en C) et leur utilisation (en OCaml.)
- L'implantation du réseau de neurones devra utiliser les aspects objets d'OCaml.
- Enfin, le partie principale (gestion de la ligne de commande et/ou lancement d'une interface graphique) du programme est à réaliser en OCaml.
- Dans la mesure du possible, l'interface graphique devra être autonome et exécutée dans un thread différent de la partie traitement.

Ces recommandations sont strictes, mais dans le cas où vous voudriez choisir un découpage différent, ce choix devra être clairement précisé et justifié. Les justifications acceptables pour remplacer une partie en OCaml par une partie en C sont :

- Si vous avez besoin d'effectuer des manipulations directs de la mémoire (`mmap, sbrk, ...`)
- Si vous avez besoin d'utiliser une bibliothèque ne disposant pas d'une version pour OCaml.

Remarque : Il n'y a pas de bonne justification pour remplacer une partie en C par une partie en OCaml.

Attention : Ces modifications au découpage recommandé doivent être minimales. Si vous pouvez isoler la partie en C et réaliser une interface d'accès (un stub) *en OCaml, vous devez le faire. Notamment, pour l'usage de bibliothèque non disponible en OCaml, nous vous conseillons de créer un* stub *(interface C/OCaml) pour la bibliothèque ou tout du moins pour la partie utilisée.*

Nous vous conseillons de découper chaque partie dans un module autonome pouvant être remplacé. Pour ce faire vous devrez établir une interface de communication claire (structures des données et fonctions d'accès simple sans "aller-retour" avec le module.) La partie chargement/pré-traitement disposera d'une interface en OCaml, mais sera implanté en C.

La ligne de commande de votre programme doit fournir au moins les options `-help` et `-help` qui affichent une aide succincte (brève description, format de la ligne de commande et liste des options accompagnées d'une brève description.) Si au moins une option est nécessaire votre programme affichera également l'aide, s'il est appeler sans option.

4 Calendrier des soutenances

Comme vous devez être quatre par groupe, le calendrier est découpé en quatre, mais ce n'est qu'à titre indicatif. Ce qui importe ce sont les tâches qui doivent être réalisées pour chaque soutenance.

1ère soutenance	Les différentes méthodes de rotation d'image et de gommage de bruits de fond.	Détection de l'angle de rotation d'une image pour la redresser, une première application de gommage de bruits de fond.	Première version du pré-traitement de l'image.	ébauche de l'interface graphique.
2ème soutenance	Présentation de la détection du positionnement des lignes à partir d'un fichier image "parfait".	Détection du positionnement des caractères et découpage.	Mise en relations des modules du projet. Travail sur la mise en page.	Site Web. Avancement de l'interface graphique.
3ème soutenance	Recherche sur l'apprentissage et sur la représentation des images.	Développement des bases du module chargé de la reconnaissance des symboles.	Intégration du module de reconnaissance des symboles dans l'application, et gestion des symboles non reconnus.	Interface graphique complète.

Description des pièces à fournir

A chaque soutenance vous devez fournir :
- Un plan de soutenance,
- Un rapport de soutenance,
- Un site Web.

A la soutenance finale vous devez fournir :
- Un plan de soutenance,
- Un rapport de projet,
- Un dossier d'exploitation,
- Le projet,
- Un site Web,
- Une procédure d'installation/Désinstallation.

Un plan de soutenance (à chaque soutenance)

Il présente sur une page et sous la forme d'un plan, la façon dont va se dérouler votre soutenance. Cela sous-entend évidemment une préparation de celle-ci (ne serait-ce que bien gérer votre temps). Il doit être remis au début de chaque soutenance.

Un rapport de soutenance (à chaque soutenance)

Il présente sur une vingtaine de pages (minimum) ce qui a été réalisé depuis la dernière soutenance ou depuis la validation du cahier des charges pour la première soutenance. Sa présentation [1] peut-être calquée sur celle du rapport de projet présenté plus loin. Il doit être remis au début de chaque soutenance. Il doit faire le bilan de ce qui a été fait (avances et/ou retards sur planning), par qui et présenter ce qui doit être fait pour la prochaine fois

[1] Toute présentation, qu'elle soit écrite ou orale, doit contenir un plan ainsi qu'une introduction et une conclusion.

Un site Web (à chaque soutenance)

Celui-ci doit contenir une page d'accueil permettant d'accéder aux différents éléments suivants :

- Une présentation du projet (l'historique, les membres, la chronologie de réalisation, les problèmes rencontrés, les solutions envisagées),
- Les liens sur les sites (des membres, des logiciels, images, sons, librairies, applets et autres éléments que vous auriez pu utiliser),
- Un download du rapport, du projet et d'une version lite de celui-ci (sans toutes les choses inutiles à l'exécution du projet).

Un rapport de projet (à la dernière soutenance)

Il est à remettre lors de la dernière soutenance, il doit contenir 50 pages minimum annexes non comprises (les sources du projet sont fournies hors rapport). Une structure [1] acceptable pourrait être :

- Reprise du cahier des charges,
- Plusieurs présentations possibles :
  - Chronologique (groupe),
  - Chronologique (individuelle),
  - Individuelles (répartition des tâches),
  - Autres.
- Récit de la réalisation :
  - Ses joies,
  - Ses peines,
  - Etc.
- Les annexes comprennent :
  - Les exemples d'impression,
  - les exemples d'écran,
  - les jeux d'essai,
  - les dessins d'origine,
  - Etc.

Remarques :

Quelle que soit la structure retenue pour les rapports (chronologique, individuelle, etc.), elle doit dans tous les cas décrire de manière très EXPLICITE qui a fait quoi.

Un Dossier d'exploitation (à la dernière soutenance)

Le dossier d'exploitation doit contenir les éléments suivants :

- Manuel d'installation,
- Manuel d'utilisation [1].

Le projet (à la dernière soutenance)

Le projet (un exécutable) doit être accompagné des éléments suivants :

- Les sources,
- Les différentes bibliothèques utilisées par votre projet,

Une procédure d'installation/Désinstallation (à la dernière soutenance)

Celle-ci doit permettre d'installer le projet, le site et tous les éléments qui les accompagnent. Bien sûr, le choix des éléments à installer doit être possible et il va de soit qu'une procédure de désinstallation est fournie.

[1] Une aide contextuelle doit être intégrée aux exécutables du projet